タブー探索による直交ラテン方陣の構成(連続と離散の最適化数理)

AUTHOR(S):

土村, 展之; 茨木, 俊秀

CITATION:

土村, 展之 ...[et al]. タブー探索による直交ラテン方陣の構成(連続と離散の最適化数理). 数理解析研究所講究録 1997, 981: 1-10

ISSUE DATE:

1997-03

URL:

http://hdl.handle.net/2433/60902

数理解析研究所講究録
981巻 1997年 1-10

タブー探索による直交ラテン方陣の構成

京都大学工学部　土村 展之　(Nobuyuki Tsuchimura)
京都大学工学部　茨木 俊秀　(Toshihide Ibaraki)

1　はじめに

n 次の直交ラテン方陣を構成する問題は, 古来難しい組合せ問題の一つとして知られている. オイラーはすべての $n \equiv 2 \pmod 4$ に対して n 次の直交ラテン方陣は存在しないと予想したが, その後10次について存在が証明され, さらに $n = 2, 6$ を除いてすべての n に対し存在することが判明した [4, 5]. しかし, これは直交ラテン方陣の一つの構成法が得られたというだけで, これら以外の直交ラテン方陣も存在することが知られているが, 全体で何個存在するかなどの正確な様子は未解決である. いろいろな直交ラテン方陣を求める一つの方法として, ランダムに与えられた初期解から, それを修正することによって直交ラテン方陣を構成することが考えられるが, 一般に極めて困難な組合せ問題である.

一方, 困難な組合せ問題に対し, メタヒューリスティックスと呼ばれる枠組みに従った種々の近似アルゴリズムが注目を集め, 多くの成功例が報告されている. その中でもタブー探索は, 複雑な条件であっても, それらを満たす解を見出す能力が高いと言われている. そこで, 直交ラテン方陣の構成問題に対してタブー探索にもとづくアルゴリズムを作り, その探索能力を評価するとともに, 効果的なタブー探索を実施するための方針を得ることを目指す.

2　定義・準備

2.1　直交ラテン方陣

n 次のラテン方陣は, n 行 n 列の2次元のセルの中に, ラベル $i \in N = \{1, 2, ..., n\}$ をならべて, どの行・どの列を見ても, 同じラベルが2つ以上存在しないものをいう. 各ラベル $i \in N$ はちょうど n 回ずつ登場する. 直交ラテン方陣は, n (≥ 2) 次のラテン方陣を2枚作り, 2枚の同じセルにあるラベル2つを順序対と見るとき, これら n^2 個の順序対がすべて異なるものをいう.

n 次の直交ラテン方陣は, $n = 2, 6$ を除いてすべての n に対し存在する. 特に, n が奇数 (図1) か4の倍数ならば簡単に構成できる. $n \equiv 2 \pmod 4$ の場合の構成法も知られている [4].

1	2	3	4	5
2	3	4	5	1
3	4	5	1	2
4	5	1	2	3
5	1	2	3	4

1	2	3	4	5
5	1	2	3	4
4	5	1	2	3
3	4	5	1	2
2	3	4	5	1

→ 順序対

(1,1)	(2,2)	(3,3)	(4,4)	(5,5)
(2,5)	(3,1)	(4,2)	(5,3)	(1,4)
(3,4)	(4,5)	(5,1)	(1,2)	(2,3)
(4,3)	(5,4)	(1,5)	(2,1)	(3,2)
(5,2)	(1,3)	(2,4)	(3,5)	(4,1)

図 1: 5 次の直交ラテン方陣

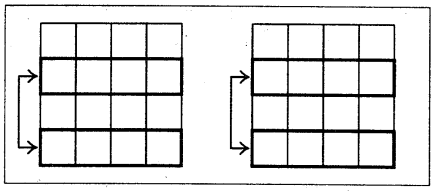


図 2: 行の交換

1	2	**3**	**4**
2	**3**	**4**	1
3	**4**	1	2
4	1	2	**3**

→
(3 と 4 を置換)

1	2	**4**	**3**
2	**4**	**3**	1
4	**3**	1	2
3	1	2	**4**

図 3: ラベルの置換

2.2 直交ラテン方陣の表現

2 枚のラテン方陣の同じ 2 つの行 (列) を入れ換える操作を施しても, 直交ラテン方陣の条件には影響がなく, 表面的には異なる直交ラテン方陣を作ることができる (図 2). また, 直交ラテン方陣のすべてのラベル $1, 2, ..., n$ に任意の置換を施して書き改めても, 直交ラテン方陣が得られる (図 3). この時, 2 枚のラテン方陣に異なる置換を施してもよい.

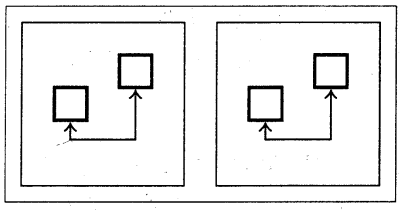
この 2 つの性質から, 直交ラテン方陣は, 2 枚のラテン方陣の第 1 行のラベルを $1, 2, ..., n$ に, さらに 1 枚の第 1 列を $1, 2, ..., n$ に正規化することができる (図 1).

2.3 タブー探索

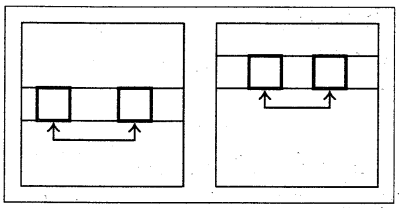
タブー探索は, 局所探索に基づくメタヒューリスティックスの一種である [1, 2, 3]. 解を保持しながら, それを近傍の中で最も良い解へ更新するという操作を反復する (局所探索) が, この際, タブーリストを用いて解の禁止領域を作りサイクリングを防ぎつつ, 局所最適解から抜け出そうとするのが特徴である. タブーリストは, 一般に解そのものではなく, 前の解との関係をあらわす変化で記憶するのが良いとされている. さらに, 長期メモリに探索過程の記録を残し, これを利用することで探索方向を分散させることも提案されている.

3 タブー探索の適用方法

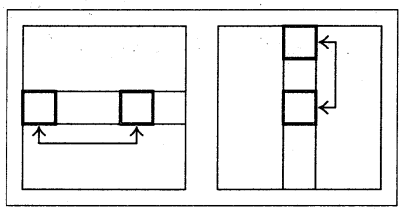
直交ラテン方陣は決定問題なので, 探索解において満たされていない条件の数を目的関数として持たせ, 最適化問題に帰着させることでタブー探索を適用した. 目的関数値が 0 になれば組合せ問題が解けたことになる. タブー探索の実行にあたって, 各部分の構成にはいろいろな可能性が考えられる. 以下, 本実験でどのような方法を試みたかを述べる.



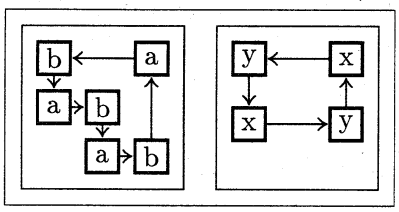
[直交条件 (a)]
任意の 2 つの順序対の交換



[行条件 (a)]
ある 1 行の 2 つのラベルの交換



[行条件か列条件]
ある 1 行か 1 列の 2 つのラベルの交換



[行条件と列条件]
一連のラベルの交換

図 4: 探索解の制限と近傍

3.1 目的関数

直交ラテン方陣には, 行条件 (1 行に各ラベルが 1 回ずつ現れる)・列条件 (1 列に各ラベルが 1 回ずつ現れる)・直交条件 (各順序対が 1 回ずつ現れる) の 3 種類の制約条件がある. まず, 2 枚のラテン方陣の各行および各列に対し, 現れていない N のラベルの数を数える. また, $N \times N$ の順序対のうち, 現れていないものの数を数える. そして, これらの数の合計をとり, 目的関数とした.

3.2 探索解の制限と近傍

探索解の構成において, 2 枚の $n \times n$ 行列の $2n^2$ 個のセルのそれぞれに任意のラベルを許し, 直交ラテン方陣の 3 種類の制約条件を目的関数として処理するのは, ある意味で最も自然なアプローチであろう. しかしこの方法では探索空間の自由度が大きすぎるため, 制約を満たす解を求めるまでに多数回の反復を要し, 有効な探索が困難であるとも考えられる. そこで, 特定の制約を満たす解のみに探索を限定する方法も試みた (図 4 参照).

- **[条件なし]** 探索解の範囲を限定しない. 近傍は, 現在の探索解の任意の 1 つのセルにあるラベルを, 別のラベルに変えて得られる解集合とする.

- 直交条件を満たすものに探索解の範囲を限定する. この場合, 解は $N \times N$ の順序対に対する置換とみなすことができる.
  [直交条件 (a)] 近傍は, 任意の 2 つの順序対の位置の交換により得られる解集合とする.
  [直交条件 (b)] 近傍を, 交換する順序対が同一行か同一列にある場合に制限する.

- 行条件を満たすものに探索解の範囲を限定する. この場合, 各行は N の順列となる.
  [行条件 (a)] 近傍は, ある 1 行の 2 つのラベルの交換により得られる解集合とする.
  [行条件 (b)] 上の近傍に, ある 1 行の 2 つの順序対の交換により得られる解集合も加える.

- [**行条件か列条件**] 2 枚の行列の, 1 枚は行条件を, もう 1 枚は列条件を満たすものに探索解の範囲を限定する. 近傍は, ある 1 行 (あるいは 1 列) の 2 つのラベルの交換により得られる解集合とする.

- [**行条件と列条件**] 行条件と列条件を同時に満たすものに探索解の範囲を限定する. この場合, 探索解は常に 2 枚のラテン方陣を与えるので, 直交条件を満たす解を探索することになる. 近傍は, 以下の操作によって生成され得る解集合とする. 任意のセル P_0 のラベル a を別のラベル b に変える. すると, P_0 と同じ行にラベル b を持つセル P_1 が存在するので, このラベルを a に変える (行操作). 次に, P_1 と同じ列にラベル a を持つセル P_2 が存在するので, このラベルを b に変える (列操作). 以下, 行条件と列条件を同時に満たすまで, 行操作と列操作を交互に繰り返す.

3.3 タブーリストの持ち方

タブーリストは, すでに探索した解に戻ることを防ぐための解の禁止領域である. 普通は, 逆戻りの動きを禁止するだけで十分な働きをするので, タブーリストは前の解との変化を記憶するのが良いとされている. 本実験では, 以下の方法を検討した.

- [**セル**] ラベルが変化したセルを覚え, 一定期間そのセルのラベルを変更しない.

- [**ラベル**] ラベルが変化したセルと, 変化前のラベルを覚え, 一定期間そのセルに元のラベルを戻さない.

- [**セル対**] 交換した 2 つのラベルのセル対を覚え, 同じ 2 つのセルのラベルを交換しない.

- [**最近の解**] 最近の解そのものを覚え, 同じ解に戻ることを禁止.

- [**過去の解 (1)**] 過去の解そのものをすべて覚え, 同じ解に戻ることを禁止.

- [**過去の解 (2)**] 過去の解そのものをすべて覚え, 2 回続けて過去の解に戻ることを禁止. (ただし, 直前の解には戻らない.)

- [**更新前後の解**] 解の更新の際の, 前後の解を過去すべてについて記憶し, 同じ前後解への更新を禁止. (前後関係の入れ代わる場合も禁止.)

3.4 近傍の削減

近傍探索では, 現在の探索解の制約違反に直接かかわるラベルを含むような交換操作のみに探索を限定して, 計算時間を短くする工夫をした. [**行条件 (a)**] [**行条件 (b)**] [**行条件か列条件**] の場合は, 各列について重複するラベルを持つセルと, 重複する順序対を持つセルを調べ上げ, これらのセルを含む交換のみを探索する. [**直交条件 (a)**] [**直交条件 (b)**] の場合は, 各行と各列について重複するラベルを持つセルを調べ上げ, そのようなセルどうしの交換のみを探索する.

3.5 目的関数値の評価変更

最適解の近傍内に存在する解の目的関数値は 4 以上なので, 目的関数値が 2 の解は最適解に近いとは限らない. そこで, 目的関数の評価を, ある変形を通して行ってみた (目的関数値 2 を 5 と評価する, など).

3.6 その他

近傍の探索方法に, 現在の解を改善する解を見つければただちに探索を打ち切ってその近傍へ移る first 改善と, すべての近傍を探索したのち最も良い解に移る best 改善の 2 つの方法がある. 本実験では best 改善を採用した. その理由は, 探索解のうち近傍に改善解を持つものが 3~4 割しか存在せず (6 割程度は同点の解しか持たず, 3~5%は改悪解しか持たない), しかもその 7 割は改善解を 1 つしか持たないので, first 改善でもすべての近傍を探索することが多く, best 改善で組織的にすべての近傍を探索した方が計算時間を節約できるからである. また, 近傍の中に最も良い目的関数値をとる解が複数ある場合は, その中からランダムに 1 つ選ぶことにした.

タブー探索の一般的枠組みでは, 過去の探索記録を長期メモリに残し, 探索域を分散させるという方法が提案されている. この実験でも, いくつかの方法を試したが, 良い結果が得られなかったので採用しないことにした.

4 計算実験と結果分析

実験には ワークステーション Sun ULTRA 1 (167MHz) を用い, 言語は C 言語を用いた. 計算実験は, 以下の要領で行った.

- 解の更新 1 回を 1 反復と数え, 反復回数の上限を設け, 探索を打ち切る.
- 目的関数値が 0 になれば, 次の反復はランダムに作った解から始める.
- 見つけた最適解 (直交ラテン方陣) の個数と, 探索に要した計算時間を記録する.
  そして, 最適解 1 個あたりの平均計算時間を性能評価の基準とする.

探索解の制限と近傍に 7 種類, タブーリストの持ち方に 7 種類の方法を考えたが, そのうち [**行条件と列条件**] は, 初期解の生成が困難なうえ, 探索解の制限が強すぎて十分な探索ができなかったため, 以下では取り上げない. タブーリストの長さは, [**セル**], [**ラベル**], [**セル対**] では $2, 3, ..., 8$ について, [**最近の解**] では $10, 15, ..., 40$ について試み, それぞれ結果の良いもの 3 つを選んで平均をとった.

$n = 7$ について, 近傍の削減をする場合の結果を表 1, 近傍の削減をしない場合の結果を表 2 に示す. $n = 8, 9$ については, 探索能力の高いものを表 3 と 4 に抜粋した. なお, $n = 9$ では見つけた最適解の個数が極端に少ないため, 平均計算時間に大きなばらつきが生じて性能を反映しないので, 目的関数値が 2 以下の解について数えた.

4.1 近傍の削減

表 1 と 2 を比較することにより, 近傍の削減を行うと, 見つけた最適解の個数は若干減るが, 計算時間が 1/2~1/10 に減り, 最適解 1 個あたりの平均計算時間は大きく改善していることがわかる. これより近傍削減の効果は大きいと言える.

なお, これとは別に, 正規化を利用して 2 つの行列の第 1 行のラベルを $1, 2, ..., n$ に固定する方法による近傍削減も試みたが, こちらは逆効果であった.

4.2 探索解の制限と近傍

探索解の制限と近傍に関して, [条件なし] と [直交条件 (b)] は探索能力が低い. これは, 前者は探索解の制限が弱すぎるため, 後者は近傍が小さすぎるためと考えられる.

[行条件 (b)], [行条件 (a)], および [行条件か列条件] は性能が良く, [直交条件 (a)] はやや劣る. 前者 3 つは, 探索解の制限方法が似通っているうえに制限の強さは等しいが, 性能にはやや差があり, 探索能力の高いものから並べるとこの順番になる. このような性能差があらわれる原因として, 以下のことが考えられる. 順序対の交換を, [行条件 (b)] では近傍操作 1 回で, [行条件 (a)] では 2 回で実現できるが, [行条件か列条件] では実現できない. すなわち, 順序対の交換がしやすいほど探索能力が高いと言える.

4.3 タブーリストの持ち方

タブーリストの持ち方では, [セル] が非常に悪く, また [ラベル] と [セル対] は, $n = 7$ の時にはあまり差があらわれないが, $n = 8$ のときには解を直接記憶する 4 つの方法に比べてかなり性能が劣ることがわかる (表 3). 近傍の禁止領域の大きさは, [セル] が一番大きく, 次いで [ラベル] と [セル対], さらに残りの 4 つという順序になるが, 上述の結果より, 近傍の禁止領域が狭いほど探索能力が高いので, この問題に関しては, 解を緻密に探索するのが良いと言える. なお, [セル], [ラベル], および [セル対] の 3 つには, タブーリストの長さが変わると探索能力が大きく影響を受けるという欠点もある.

解を直接記憶する 4 つの方法を比べると, [最近の解], [過去の解 (2)], [更新前後の解] の 3 つは同程度の性能であるが, [過去の解 (1)] はこれらより若干劣る. このことから, 単純に過去に探索した解への動きを禁止するのは良くないようである.

なお, [過去の解 (1)], [過去の解 (2)], [更新前後の解] は, 過去に探索した解をすべて記憶するため, 非常に大きな記憶領域を必要とする. 一方, [最近の解] は, 探索能力はこれらとほぼ同等でありながら, パラメータで定めた記憶領域しか用いないので, より実用的である. さらに [最近の解] は, タブーリストの長さの変化による探索能力への影響が少ないので, 取り扱う上で便利である. (本実験で解の記憶はハッシュ関数を用いて近似的に実現している.)

4.4 その他

結果は示さないが, 目的関数値の評価変更は, 目的関数値 2~4 を 2~5 に対応させる様々な写像を試みたが, 探索能力に影響はなく, 効果がないことがわかった. これは, 目的関数値 2 の解が 4 の解に比べてかなり少ないことが原因と考えられる.

長期メモリについても, 効果のある方法は見つけられなかった. この問題は, 制約が対称的で, どのラベルも目的関数に同じ影響力を持つため, 特に探索域を分散させなくても十分な探索が行えるので, 長期メモリの効果が表れにくいものと思われる.

最後に, $n = 9$ で最も成績の良かった [行条件 (b)] と [更新前後の解] の組合せと, 記憶容量の観点からより実用的と考えられる [行条件 (b)] と [最近の解] の組合せで $n = 10$ について探索を行い, どちらも 1 つずつ直交ラテン方陣を構成することに成功した (図 5, 6). これらは, 行の交換などの様々な変形を施しても, 文献 [4] によるもの (図 7) とは一致しないことを確認した.

表 1: 見つけた最適解の個数と計算時間, および最適解 1 個あたりの平均計算時間 ($n = 7$, 反復 50 万回, 近傍の削減をする).

タブーリストの持ち方	探索解の制限と近傍 [条件なし] 最適解の個数	計算時間 (秒)	平均時間 (秒/個)	[直交条件 (a)] 最適解の個数	計算時間 (秒)	平均時間 (秒/個)	[直交条件 (b)] 最適解の個数	計算時間 (秒)	平均時間 (秒/個)
[セル]	2.0	46	23	1.7	191	115	0.3	196	589
[ラベル]	0.0	53	∞	5.7	156	28	0.7	198	296
[セル対]	—	—	—	12	166	13	0.3	188	563
[最近の解]	2.3	51	22	14	130	9.0	1.0	200	200
[過去の解 (1)]	2.3	48	21	15	162	11	0.3	208	624
[過去の解 (2)]	1.3	33	24	12	128	10	0.7	197	296
[更新前後の解]	0.0	55	∞	7.3	92	13	0.0	192	∞

タブーリストの持ち方	探索解の制限と近傍 [行条件か列条件] 最適解の個数	計算時間 (秒)	平均時間 (秒/個)	[行条件 (a)] 最適解の個数	計算時間 (秒)	平均時間 (秒/個)	[行条件 (b)] 最適解の個数	計算時間 (秒)	平均時間 (秒/個)
[セル]	3.3	109	33	4.7	108	23	9.3	156	17
[ラベル]	6.3	100	16	16	103	6.6	16	139	8.9
[セル対]	8.0	109	14	12	97	8.1	18	140	7.9
[最近の解]	15	111	7.3	18	104	5.8	28	146	5.3
[過去の解 (1)]	8.7	117	13	11	112	10	20	152	7.7
[過去の解 (2)]	9.3	104	11	13	102	7.6	19	136	7.1
[更新前後の解]	12	105	8.8	17	104	6.0	18	134	7.4

5 まとめ

直交ラテン方陣を構成する問題に対して, メタヒューリスティックスの 1 つであるタブー探索の適用を試みた. その結果, 非常に困難な組合せ問題と考えられる $n = 10$ に対する直交ラテン方陣をいくつか求めることに成功した. このことから, タブー探索は, 複雑な組合せ問題を解くための有望なツールの 1 つであるといえる.

この問題に関して, タブー探索の構成方法を種々試みた結果, 効果的なアルゴリズムを構成するための指針として, 以下のことが観測できた. まず, 探索解の制限と近傍の選択は, 探索能力を大きく左右する非常に重要な要素であった. また, タブーリストの持ち方は, 近傍内で禁止される解の数が少ないものを選べば, タブーリストの長さにもあまり影響されず, どの方法でも安定して高い性能が得られた. 中でも, 最近探索した解に戻ることを禁止するのは, ハッシュ関数を用いれば必要とする記憶領域も少なく, 非常に効果的であった. なお, タブー探索の基本的戦略の一つである長期メモリは, 本問題に対しては, 効果が認められなかった.

表 2: 見つけた最適解の個数と計算時間, および最適解 1 個あたりの平均計算時間 ($n = 7$, 反復 50 万回, 近傍の削減をしない).

タブーリストの持ち方	探索解の制限と近傍 [条件なし] 最適解の個数	計算時間 (秒)	平均時間 (秒/個)	[直交条件 (a)] 最適解の個数	計算時間 (秒)	平均時間 (秒/個)	[直交条件 (b)] 最適解の個数	計算時間 (秒)	平均時間 (秒/個)
[セル]	3.7	181	49	2.0	1490	746	0.3	335	1010
[ラベル]	0.0	181	∞	11	1520	134	0.0	336	∞
[セル対]	—	—	—	23	1490	64	0.3	341	1020
[最近の解]	5.0	214	43	25	1610	65	1.3	388	291
[過去の解 (1)]	0.7	196	295	16	1630	102	0.0	366	∞
[過去の解 (2)]	3.0	195	65	25	1630	66	2.3	367	157
[更新前後の解]	0.0	203	∞	23	1660	71	0.3	382	1150

タブーリストの持ち方	探索解の制限と近傍 [行条件か列条件] 最適解の個数	計算時間 (秒)	平均時間 (秒/個)	[行条件 (a)] 最適解の個数	計算時間 (秒)	平均時間 (秒/個)	[行条件 (b)] 最適解の個数	計算時間 (秒)	平均時間 (秒/個)
[セル]	4.0	241	60	5.0	229	46	12	347	30
[ラベル]	5.7	246	43	7.7	237	31	15	350	23
[セル対]	8.0	242	30	12	227	18	18	346	19
[最近の解]	14	271	19	16	257	16	28	396	14
[過去の解 (1)]	8.0	267	33	10	254	26	21	394	18
[過去の解 (2)]	13	267	20	13	254	20	23	394	17
[更新前後の解]	12	273	22	12	263	21	21	403	20

参考文献

[1] F. Glover: "Tabu search - Part I," *ORSA Journal on Computing* **1** (1989) 190-206; Part II, ditto, **2** (1990) 4-32.

[2] 久保幹雄: 「メタヒューリスティックス」, 離散構造とアルゴリズム IV, 近代科学社 (1995) 171-222.

[3] 茨木俊秀: 「組合せ最適化の手法」, 電学論C, **114** (1994) 411-419.

[4] Hall, M. Jr: *Combinatorial theory*, Blaisdell Co., Waltham, Mass. (1967). (岩堀 信子 訳: 組合せ理論, 吉岡書店 (1971).)

[5] C. L. Liu: *Introduction to combinatorial mathematics*, McGrew-Hill Book Company (1968). (伊理 正夫, 伊理 由美 共訳: 組合せ数学入門 II, 共立全書 (1972).)

表 3: 見つけた最適解の個数と計算時間, および最適解 1 個あたりの平均計算時間 ($n = 8$, 反復 200 万回, 近傍の削減をする).

タブーリストの持ち方	探索解の制限と近傍								
	[直交条件 (a)]			[行条件 (a)]			[行条件 (b)]		
	最適解の個数	計算時間 (秒)	平均時間 (秒/個)	最適解の個数	計算時間 (秒)	平均時間 (秒/個)	最適解の個数	計算時間 (秒)	平均時間 (秒/個)
[セル]	0.0	968	∞	1.3	585	439	1.7	831	499
[ラベル]	1.0	1180	1180	2.0	552	276	4.3	784	181
[セル対]	1.7	874	525	2.3	531	228	3.3	810	243
[最近の解]	5.7	821	145	6.3	589	93	9.0	827	92
[過去の解 (1)]	3.3	963	289	3.7	621	169	6.7	885	133
[過去の解 (2)]	3.7	838	229	5.7	546	96	7.3	761	104
[更新前後の解]	4.3	677	156	6.0	562	94	10	748	75

表 4: 見つけた最適解の個数と計算時間, および最適解 1 個あたりの平均計算時間 ($n = 9$, 反復 800 万回, 近傍の削減をする).

タブーリストの持ち方	探索解の制限と近傍								
	[直交条件 (a)]			[行条件 (a)]			[行条件 (b)]		
	最適解† の個数	計算時間 (秒)	平均時間 (秒/個)	最適解† の個数	計算時間 (秒)	平均時間 (秒/個)	最適解† の個数	計算時間 (秒)	平均時間 (秒/個)
[最近の解]	20	4910	245	23	2930	126	41	4220	103
[過去の解 (2)]	24	4410	184	23	2880	127	49	4070	84
[更新前後の解]	44	3480	78	30	3040	101	54	4020	74

† 最適解ではなく, 目的関数値 2 以下の解の個数

3	10	8	7	2	6	5	1	4	9
5	4	7	2	6	1	3	10	9	8
4	2	9	8	3	5	1	6	10	7
2	7	5	3	8	9	6	4	1	10
8	6	1	9	5	4	10	2	7	3
7	3	4	1	9	10	2	8	6	5
9	1	3	10	7	2	4	5	8	6
10	8	2	6	1	3	9	7	5	4
6	9	10	5	4	7	8	3	2	1
1	5	6	4	10	8	7	9	3	2

8	3	4	6	10	7	9	1	2	5
4	10	8	5	2	9	1	7	6	3
1	4	2	8	3	10	7	6	5	9
3	2	5	7	6	4	8	9	10	1
7	5	3	9	8	6	4	2	1	10
5	9	7	4	1	8	6	10	3	2
10	8	6	2	7	1	5	3	9	4
6	1	9	10	5	2	3	4	7	8
9	7	10	1	4	3	2	5	8	6
2	6	1	3	9	5	10	8	4	7

図 5: 本実験による 10 次の直交ラテン方陣 (1)

6	9	2	10	8	3	4	1	7	5
8	7	9	5	3	4	1	10	2	6
9	1	8	3	4	6	2	7	5	10
3	2	10	8	7	5	9	4	6	1
10	6	3	1	9	7	5	2	8	4
7	10	4	9	5	8	3	6	1	2
4	3	7	2	6	1	8	5	10	9
2	5	6	7	1	9	10	8	4	3
1	4	5	6	2	10	7	9	3	8
5	8	1	4	10	2	6	3	9	7

3	7	8	4	9	2	1	5	10	6
1	4	10	5	7	6	8	2	3	9
2	6	3	9	10	8	5	1	4	7
5	9	6	2	3	7	4	8	1	10
8	10	4	1	5	9	3	7	6	2
7	3	9	6	8	5	10	4	2	1
4	1	2	10	6	3	7	9	5	8
6	2	5	8	4	1	9	10	7	3
9	5	1	7	2	10	6	3	8	4
10	8	7	3	1	4	2	6	9	5

図 6: 本実験による 10 次の直交ラテン方陣 (2)

1	7	6	5	10	9	8	2	3	4
8	2	1	7	6	10	9	3	4	5
9	8	3	2	1	7	10	4	5	6
10	9	8	4	3	2	1	5	6	7
2	10	9	8	5	4	3	6	7	1
4	3	10	9	8	6	5	7	1	2
6	5	4	10	9	8	7	1	2	3
3	4	5	6	7	1	2	8	9	10
5	6	7	1	2	3	4	9	10	8
7	1	2	3	4	5	6	10	8	9

1	8	9	10	2	4	6	3	5	7
7	2	8	9	10	3	5	4	6	1
6	1	3	8	9	10	4	5	7	2
5	7	2	4	8	9	10	6	1	3
10	6	1	3	5	8	9	7	2	4
9	10	7	2	4	6	8	1	3	5
8	9	10	1	3	5	7	2	4	6
2	3	4	5	6	7	1	8	9	10
3	4	5	6	7	1	2	10	8	9
4	5	6	7	1	2	3	9	10	8

図 7: 文献 [4] による 10 次の直交ラテン方陣の例